

"EXPRESS MAIL" Mailing Label No....EL 851565960 US.....
Date of Deposit.....APRIL 25, 2001.....

SYSTEM AND METHOD FOR SELECTIVELY AND AUTOMATICALLY
MODIFYING THE SOURCE CODE OF A COMPUTER PROGRAM

5 BACKGROUND OF THE INVENTION

Technical Field of the Invention

50
55
60
65
70
75
80
85
90
95
100
105
110
115
120
125
130
135
140
145
150
155
160
165
170
175
180
185
190
195
200
205
210
215
220
225
230
235
240
245
250
255
260
265
270
275
280
285
290
295
300
305
310
315
320
325
330
335
340
345
350
355
360
365
370
375
380
385
390
395
400
405
410
415
420
425
430
435
440
445
450
455
460
465
470
475
480
485
490
495
500
505
510
515
520
525
530
535
540
545
550
555
560
565
570
575
580
585
590
595
600
605
610
615
620
625
630
635
640
645
650
655
660
665
670
675
680
685
690
695
700
705
710
715
720
725
730
735
740
745
750
755
760
765
770
775
780
785
790
795
800
805
810
815
820
825
830
835
840
845
850
855
860
865
870
875
880
885
890
895
900
905
910
915
920
925
930
935
940
945
950
955
960
965
970
975
980
985
990
995
1000
1005
1010
1015
1020
1025
1030
1035
1040
1045
1050
1055
1060
1065
1070
1075
1080
1085
1090
1095
1100
1105
1110
1115
1120
1125
1130
1135
1140
1145
1150
1155
1160
1165
1170
1175
1180
1185
1190
1195
1200
1205
1210
1215
1220
1225
1230
1235
1240
1245
1250
1255
1260
1265
1270
1275
1280
1285
1290
1295
1300
1305
1310
1315
1320
1325
1330
1335
1340
1345
1350
1355
1360
1365
1370
1375
1380
1385
1390
1395
1400
1405
1410
1415
1420
1425
1430
1435
1440
1445
1450
1455
1460
1465
1470
1475
1480
1485
1490
1495
1500
1505
1510
1515
1520
1525
1530
1535
1540
1545
1550
1555
1560
1565
1570
1575
1580
1585
1590
1595
1600
1605
1610
1615
1620
1625
1630
1635
1640
1645
1650
1655
1660
1665
1670
1675
1680
1685
1690
1695
1700
1705
1710
1715
1720
1725
1730
1735
1740
1745
1750
1755
1760
1765
1770
1775
1780
1785
1790
1795
1800
1805
1810
1815
1820
1825
1830
1835
1840
1845
1850
1855
1860
1865
1870
1875
1880
1885
1890
1895
1900
1905
1910
1915
1920
1925
1930
1935
1940
1945
1950
1955
1960
1965
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
2025
2030
2035
2040
2045
2050
2055
2060
2065
2070
2075
2080
2085
2090
2095
2100
2105
2110
2115
2120
2125
2130
2135
2140
2145
2150
2155
2160
2165
2170
2175
2180
2185
2190
2195
2200
2205
2210
2215
2220
2225
2230
2235
2240
2245
2250
2255
2260
2265
2270
2275
2280
2285
2290
2295
2300
2305
2310
2315
2320
2325
2330
2335
2340
2345
2350
2355
2360
2365
2370
2375
2380
2385
2390
2395
2400
2405
2410
2415
2420
2425
2430
2435
2440
2445
2450
2455
2460
2465
2470
2475
2480
2485
2490
2495
2500
2505
2510
2515
2520
2525
2530
2535
2540
2545
2550
2555
2560
2565
2570
2575
2580
2585
2590
2595
2600
2605
2610
2615
2620
2625
2630
2635
2640
2645
2650
2655
2660
2665
2670
2675
2680
2685
2690
2695
2700
2705
2710
2715
2720
2725
2730
2735
2740
2745
2750
2755
2760
2765
2770
2775
2780
2785
2790
2795
2800
2805
2810
2815
2820
2825
2830
2835
2840
2845
2850
2855
2860
2865
2870
2875
2880
2885
2890
2895
2900
2905
2910
2915
2920
2925
2930
2935
2940
2945
2950
2955
2960
2965
2970
2975
2980
2985
2990
2995
3000
3005
3010
3015
3020
3025
3030
3035
3040
3045
3050
3055
3060
3065
3070
3075
3080
3085
3090
3095
3100
3105
3110
3115
3120
3125
3130
3135
3140
3145
3150
3155
3160
3165
3170
3175
3180
3185
3190
3195
3200
3205
3210
3215
3220
3225
3230
3235
3240
3245
3250
3255
3260
3265
3270
3275
3280
3285
3290
3295
3300
3305
3310
3315
3320
3325
3330
3335
3340
3345
3350
3355
3360
3365
3370
3375
3380
3385
3390
3395
3400
3405
3410
3415
3420
3425
3430
3435
3440
3445
3450
3455
3460
3465
3470
3475
3480
3485
3490
3495
3500
3505
3510
3515
3520
3525
3530
3535
3540
3545
3550
3555
3560
3565
3570
3575
3580
3585
3590
3595
3600
3605
3610
3615
3620
3625
3630
3635
3640
3645
3650
3655
3660
3665
3670
3675
3680
3685
3690
3695
3700
3705
3710
3715
3720
3725
3730
3735
3740
3745
3750
3755
3760
3765
3770
3775
3780
3785
3790
3795
3800
3805
3810
3815
3820
3825
3830
3835
3840
3845
3850
3855
3860
3865
3870
3875
3880
3885
3890
3895
3900
3905
3910
3915
3920
3925
3930
3935
3940
3945
3950
3955
3960
3965
3970
3975
3980
3985
3990
3995
4000
4005
4010
4015
4020
4025
4030
4035
4040
4045
4050
4055
4060
4065
4070
4075
4080
4085
4090
4095
4100
4105
4110
4115
4120
4125
4130
4135
4140
4145
4150
4155
4160
4165
4170
4175
4180
4185
4190
4195
4200
4205
4210
4215
4220
4225
4230
4235
4240
4245
4250
4255
4260
4265
4270
4275
4280
4285
4290
4295
4300
4305
4310
4315
4320
4325
4330
4335
4340
4345
4350
4355
4360
4365
4370
4375
4380
4385
4390
4395
4400
4405
4410
4415
4420
4425
4430
4435
4440
4445
4450
4455
4460
4465
4470
4475
4480
4485
4490
4495
4500
4505
4510
4515
4520
4525
4530
4535
4540
4545
4550
4555
4560
4565
4570
4575
4580
4585
4590
4595
4600
4605
4610
4615
4620
4625
4630
4635
4640
4645
4650
4655
4660
4665
4670
4675
4680
4685
4690
4695
4700
4705
4710
4715
4720
4725
4730
4735
4740
4745
4750
4755
4760
4765
4770
4775
4780
4785
4790
4795
4800
4805
4810
4815
4820
4825
4830
4835
4840
4845
4850
4855
4860
4865
4870
4875
4880
4885
4890
4895
4900
4905
4910
4915
4920
4925
4930
4935
4940
4945
4950
4955
4960
4965
4970
4975
4980
4985
4990
4995
5000
5005
5010
5015
5020
5025
5030
5035
5040
5045
5050
5055
5060
5065
5070
5075
5080
5085
5090
5095
5100
5105
5110
5115
5120
5125
5130
5135
5140
5145
5150
5155
5160
5165
5170
5175
5180
5185
5190
5195
5200
5205
5210
5215
5220
5225
5230
5235
5240
5245
5250
5255
5260
5265
5270
5275
5280
5285
5290
5295
5300
5305
5310
5315
5320
5325
5330
5335
5340
5345
5350
5355
5360
5365
5370
5375
5380
5385
5390
5395
5400
5405
5410
5415
5420
5425
5430
5435
5440
5445
5450
5455
5460
5465
5470
5475
5480
5485
5490
5495
5500
5505
5510
5515
5520
5525
5530
5535
5540
5545
5550
5555
5560
5565
5570
5575
5580
5585
5590
5595
5600
5605
5610
5615
5620
5625
5630
5635
5640
5645
5650
5655
5660
5665
5670
5675
5680
5685
5690
5695
5700
5705
5710
5715
5720
5725
5730
5735
5740
5745
5750
5755
5760
5765
5770
5775
5780
5785
5790
5795
5800
5805
5810
5815
5820
5825
5830
5835
5840
5845
5850
5855
5860
5865
5870
5875
5880
5885
5890
5895
5900
5905
5910
5915
5920
5925
5930
5935
5940
5945
5950
5955
5960
5965
5970
5975
5980
5985
5990
5995
6000
6005
6010
6015
6020
6025
6030
6035
6040
6045
6050
6055
6060
6065
6070
6075
6080
6085
6090
6095
6100
6105
6110
6115
6120
6125
6130
6135
6140
6145
6150
6155
6160
6165
6170
6175
6180
6185
6190
6195
6200
6205
6210
6215
6220
6225
6230
6235
6240
6245
6250
6255
6260
6265
6270
6275
6280
6285
6290
6295
6300
6305
6310
6315
6320
6325
6330
6335
6340
6345
6350
6355
6360
6365
6370
6375
6380
6385
6390
6395
6400
6405
6410
6415
6420
6425
6430
6435
6440
6445
6450
6455
6460
6465
6470
6475
6480
6485
6490
6495
6500
6505
6510
6515
6520
6525
6530
6535
6540
6545
6550
6555
6560
6565
6570
6575
6580
6585
6590
6595
6600
6605
6610
6615
6620
6625
6630
6635
6640
6645
6650
6655
6660
6665
6670
6675
6680
6685
6690
6695
6700
6705
6710
6715
6720
6725
6730
6735
6740
6745
6750
6755
6760
6765
6770
6775
6780
6785
6790
6795
6800
6805
6810
6815
6820
6825
6830
6835
6840
6845
6850
6855
6860
6865
6870
6875
6880
6885
6890
6895
6900
6905
6910
6915
6920
6925
6930
6935
6940
6945
6950
6955
6960
6965
6970
6975
6980
6985
6990
6995
7000
7005
7010
7015
7020
7025
7030
7035
7040
7045
7050
7055
7060
7065
7070
7075
7080
7085
7090
7095
7100
7105
7110
7115
7120
7125
7130
7135
7140
7145
7150
7155
7160
7165
7170
7175
7180
7185
7190
7195
7200
7205
7210
7215
7220
7225
7230
7235
7240
7245
7250
7255
7260
7265
7270
7275
7280
7285
7290
7295
7300
7305
7310
7315
7320
7325
7330
7335
7340
7345
7350
7355
7360
7365
7370
7375
7380
7385
7390
7395
7400
7405
7410
7415
7420
7425
7430
7435
7440
7445
7450
7455
7460
7465
7470
7475
7480
7485
7490
7495
7500
7505
7510
7515
7520
7525
7530
7535
7540
7545
7550
7555
7560
7565
7570
7575
7580
7585
7590
7595
7600
7605
7610
7615
7620
7625
7630
7635
7640
7645
7650
7655
7660
7665
7670
7675
7680
7685
7690
7695
7700
7705
7710
7715
7720
7725
7730
7735
7740
7745
7750
7755
7760
7765
7770
7775
7780
7785
7790
7795
7800
7805
7810
7815
7820
7825
7830
7835
7840
7845
7850
7855
7860
7865
7870
7875
7880
7885
7890
7895
7900
7905
7910
7915
7920
7925
7930
7935
7940
7945
7950
7955
7960
7965
7970
7975
7980
7985
7990
7995
8000
8005
8010
8015
8020
8025
8030
8035
8040
8045
8050
8055
8060
8065
8070
8075
8080
8085
8090
8095
8100
8105
8110
8115
8120
8125
8130
8135
8140
8145
8150
8155
8160
8165
8170
8175
8180
8185
8190
8195
8200
8205
8210
8215
8220
8225
8230
8235
8240
8245
8250
8255
8260
8265
8270
8275
8280
8285
8290
8295
8300
8305
8310
8315
8320
8325
8330
8335
8340
8345
8350
8355
8360
8365
8370
8375
8380
8385
8390
8395
8400
8405
8410
8415
8420
8425
8430
8435
8440
8445
8450
8455
8460
8465
8470
8475
8480
8485
8490
8495
8500
8505
8510
8515
8520
8525
8530
8535
8540
8545
8550
8555
8560
8565
8570
8575
8580
8585
8590
8595
8600
8605
8610
8615
8620
8625
8630
8635
8640
8645
8650
8655
8660
8665
8670
8675
8680
8685
8690
8695
8700
8705
8710
8715
8720
8725
8730
8735
8740
8745
8750
8755
8760
8765
8770
8775
8780
8785
8790
8795
8800
8805
8810
8815
8820
8825
8830
8835
8840
8845
8850
8855
8860
8865
8870
8875
8880
8885
8890
8895
8900
8905
8910
8915
8920
8925
8930
8935
8940
8945
8950
8955
8960
8965
8970
8975
8980
8985
8990
8995
9000
9005
9010
9015
9020
9025
9030
9035
9040
9045
9050
9055
9060
9065
9070
9075
9080
9085
9090
9095
9100
9105
9110
9115
9120
9125
9130
9135
9140
9145
9150
9155
9160
9165
9170
9175
9180
9185
9190
9195
9200
9205
9210
9215
9220
9225
9230
9235
9240
9245
9250
9255
9260
9265
9270
9275
9280
9285
9290
9295
9300
9305
9310
9315
9320
9325
9330
9335
9340
9345
9350
9355
9360
9365
9370
9375
9380
9385
9390
9395
9400
9405
9410
9415
9420
9425
9430
9435
9440
9445
9450
9455
9460
9465
9470
9475
9480
9485
9490
9495
9500
9505
9510
9515
9520
9525
9530
9535
9540
9545
9550
9555
9560
9565
9570
9575
9580
9585
9590
9595
9600
9605
9610
9615
9620
9625
9630
9635
9640
9645
9650
9655
9660
9665
9670
9675
9680
9685
9690
9695
9700
9705
9710
9715
9720
9725
9730
9735
9740
9745
9750
9755
9760
9765
9770
9775
9780
9785
9790
9795
9800
9805
9810
9815
9820
9825
9830
9835
9840
9845
9850
9855
9860
9865
9870
9875
9880
9885
9890
9895
9900
9905
9910
9915
9920
9925
9930
9935
9940
9945
9950
9955
9960
9965
9970
9975
9980
9985
9990
9995
10000
10005
10010
10015
10020
10025
10030
10035
10040
10045
10050
10055
10060
10065
10070
10075
10080
10085
10090
10095
10100
10105
10110
10115
10120
10125
10130
10135
10140
10145
10150
10155
10160
10165
10170
10175
10180
10185
10190
10195
10200
10205
10210
10215
10220
10225
10230
10235
10240
1

Self-measuring and self-adapting extensible kernels are also known.

Also, software debuggers are sometimes used for monitoring accesses to global variables in a computer program. Further, software simulators and specialized scripts (e.g., written in PERL) can be used in some applications for measuring the GV access statistics. Commercial products such as C-COVER are operable to insert bit flags in the source code of a computer program in order to measure which portions of the code have been exercised under a test load.

Although these solutions have been generally useful, they are beset with several deficiencies and shortcomings. For instance, whereas the use of simulators and specialized scripts can be advantageous for instrumenting a computer program in order to obtain performance statistics, resultant disk space usage is typically quite excessive. Accordingly, the amount of data that can be collected is rather too limited to be of significant value. Furthermore, instrumented simulators are very slow compared to the normal execution speed of the software, and this limits the size and complexity of the workloads that can be used to exercise the software. On the other hand, debugger tools are generally very quick, with near full execution speeds. However, they are limited to monitoring accesses to only a few GVs at a time. Similarly, the specialized scripts for instrumenting a computer program become too bulky when several GVs or function calls are to be monitored.

Additionally, with respect to operating system software, most of the extensible kernels for providing runtime instrumentation are operable with customized kernels only.

As a consequence, they are not well suited for use with real-world programs or workloads. Finally, bit flag insertion is too rudimentary a technique to provide useful instrumentation in complex computer programs.

SUMMARY OF THE INVENTION

Accordingly, the present invention advantageously provides a system and method for automatically and selectively instrumenting the source code of a computer program without these and other shortcomings and deficiencies of the current solutions. A parser is provided for scanning the source code portion to recognize select syntax structures of the source code. A code modification portion is included in the parser which inserts instrumentation code at select locations of the source code based on the modification code. The instrumented source code can be compiled and executed using test loads to exercise different portions of the source code, and thereby obtain statistical data such as the frequency of GV accesses, function calls, frequency of use of multiple code paths, etc.

In an exemplary embodiment of the present invention, the method of modifying a source code portion associated with a computer program comprises the step of scanning the entire source code portion using a parser having one or more predetermined code modification portions that are operable to insert instrumentation code at selected locations in the source code. Preferably, any computer program can be modified accordingly, which computer program may be selected from the group consisting of, for instance, an operating system kernel (e.g., HP-UX kernel), an application program and a software utility program. The computer program's

03841981-042501
102820-1864880

source code may be written in any language, e.g., in the C language. Accordingly, a C language parser such as the recursive-descent parser is preferably used, wherein the C parser is coded using the Backus-Naur Form (BNF) description of ANSI C. The predetermined code modification portions of the parser are operable to recognize select syntax structures of the source code portion and insert an instrumentation code portion at a location associated with the select syntax structure based on the content of the code modification portion of the parser. For example, counters may be instrumented at the global variable calls in the source code portion so that when the global variables are accessed during runtime, associated counters are operable to track such accesses. Similarly, a code modification portion of the parser is operable to insert instrumentation code at select locations in the source code portion in order to keep track of accesses to a particular GV from a select module of the source code.

In another exemplary embodiment, the method of modifying the source code portion associated with a computer program may also include the step of pre-processing the source code so as to remove macros, comments and other extraneous portions from the source code.

In a further aspect, the present invention is directed to a computer-readable medium operable with a processing environment, the computer-readable medium carrying a sequence of instructions which, when executed in the processing environment, causes the processing environment to perform the steps of the source code modification method summarized hereinabove. In an exemplary embodiment, the processing

environment may be comprised of a hardware architecture simulator for a target multiprocessing system.

BRIEF DESCRIPTION OF THE DRAWINGS

A more complete understanding of the present invention may be had by reference to the following Detailed Description when taken in conjunction with the accompanying drawings wherein:

FIG. 1 is a flow chart of the steps involved in an exemplary source code modification method provided in accordance with the teachings of the present invention;

FIG. 2 depicts a high level functional block diagram of an exemplary source code modification system provided in accordance with the teachings of the present invention;

FIG. 3A depicts an exemplary embodiment of a source code prior to modification;

FIG. 3B depicts an exemplary embodiment of the source code after modification; and

FIG. 4 depicts a block diagram of an exemplary multiprocessing system whose operating system code may be advantageously modified using the teachings of the present invention.

DETAILED DESCRIPTION OF THE DRAWINGS

In the drawings, like or similar elements are designated with identical reference numerals throughout the several views thereof, and the various elements depicted are not necessarily drawn to scale. Referring now to FIG. 1, depicted therein is a flow chart of the steps in an exemplary source code modification method provided in accordance with the teachings of the present invention for modifying the

source code of a computer program. Preferably, any computer program such as, for example, an operating system kernel, an application program, any software utility, et cetera, can be modified by using the teachings of the present invention.

5 Furthermore, the source code of the computer program may be written in any known or heretofore unknown computer language such as, for instance, C or C++, and the like. Accordingly, depending on the source code of the computer program, a suitable parser is provided (step 102) wherein one or more

10 predetermined code modification portions are included in accordance with the teachings of the present invention set forth in greater detail hereinbelow.

In a presently preferred exemplary embodiment of the present invention, a C language parser such as the recursive-descent parser is utilized wherein the C parser is coded

15 using the Backus-Naur Form (BNF) description of ANSI C. The recursive-descent parser is operable to provide functions to handle each syntactical structure or element of the source code language, e.g., ANSI C, which consequently permits easy

20 identification of a suitable location in the parser's source code that corresponds to the syntactical structure of the computer program being parsed. These functions are substantially simple in general and consist of the following operations: consuming tokens, determining which of several

25 possible alternative elements follows in a statement, calling other functions to handle that alternative, and returning to their caller when the handling of their own syntactical element is complete.

Because the teachings of the present invention are

30 particularly exemplified in the context of using the

recursive-descent C language parser, the general operation thereof is immediately set forth below.

In the presently preferred exemplary implementation, the C language parser makes heavy use of the language's ability to support recursive function calls, that is, a function that calls itself either directly or by calling other functions that eventually call the original function again. In the parser, this typically happens when an instance of a particular syntactical element contains other instances of the same type of syntactical element. For example, a compound statement (i.e., a statement inside the curly braces in C -- e.g., {statement}) may contain other compound statements. As can be readily appreciated by those skilled in the art, an advantageous feature of this technique is that the programming for each function only needs to handle one instance of its syntactic element. For any other instances of the syntactic element contained in this instance, the function will be called again, recursively, as necessary.

In one embodiment, the source program to be parsed is provided as a memory mapped input file. Thus, each execution of the parser reads one C language source file specified on the command line of the code. The source file is accessed by memory mapping it into the address space of the parser as a read-only code. In this instance, accordingly, the source file resides on a disk medium. A pointer returned by a specific function called *mmap()* may be used to initialize a *current position* character pointer. Various functions within the parser advance the *current position* character pointer whenever the function consumes the tokens represented by those characters. It should be recognized that by using the memory mapping technique, any function can replace "consumed"

characters by simply adjusting the *current position* backward to a previous point in the source -- typically to where the token or the syntactic element began.

One of ordinary skill in the art should appreciate that the parser functionality can be enhanced by allowing it to accept the input source file as a stream instead of requiring a disk file. In this approach, one exemplary implementation would be to have the parser recognize that the input file is a stream, copy it to a temporary disk file, operate normally, and then delete the temporary disk file.

Often in parsing a C program there are several possible syntactic elements that are legal at the current position in the source code being parsed. In order to determine which syntactic element is actually present at that position, the parser implements what are known as *Is* functions. These functions are named beginning with the word "Is" and followed by the name of the syntactic element that the function considers. If the *Is* function determines that the current syntactic element matches the name, then the *Is* function returns TRUE, otherwise it returns FALSE. For example, the *Is* function "*IsStatement()*" returns TRUE when the syntactic element at the current position of the source code being parsed is a statement and FALSE when it is not.

When an *Is* function returns to its caller, the current position within the source code must be the same as when the *Is* function was called. Most *Is* functions implement this requirement by saving the current position upon entry and restoring it just before returning. An *Is* function is allowed to modify the current position temporarily during its execution. Usually this happens because the *Is* function

needs to consume tokens or some other syntactic element in order to determine whether or not the current syntactic element is its type.

Typically, in the presently preferred exemplary embodiment of the present invention, the parser itself produces little output, except diagnostic printouts. Output is normally generated by the code that may be added to the parser in order to accomplish some additional purpose. The diagnostic printouts may be provided as a hierarchical, outline format print of the entry to and exit from each function within the parser, along with a few characters from the current position to help identify what part of the input source the parser is handling. The parser implements this functionality internally by calling a first function named *ParseEnter()* as the first executable statement within each function and by calling a second function named *ParseLeave()* as the last executable statement before the return. Preferably, parser functions are implemented such that there is a single entry and single return point, as exemplified below:

```
FunctionName()  
{  
    int      irc = 0; // Integer Return Code  
    ParseEnter("FunctionName");  
    /* Logic of the function goes here */  
EXIT;  
    ParseLeave("FunctionName");  
    Return irc;      // irc is the Integer Return Code  
}
```

Whereas the trace printout generated by such code can become quite large and the parser functionality may slow down, the resulting output can be very helpful as a diagnostic tool for the parser.

As alluded to hereinabove, the recursive-descent C parser described herein is provided with one or more predetermined code modification portions that are operable to insert appropriate instrumentation code into the source code being parsed. Preferably, the parser may be run between the C pre-processor and compiler so that it runs on input files that are free from comments, macros, and the like. Accordingly, in some implementations, it may be necessary to effectuate the step of pre-processing the source code prior to executing the parser code (step 104).

The modification code portion in the parser is preferably operable to scan the expressions in the source code for inserting the instrumentation code at the start of each expression having global variables (step 106). Based on the modification code of the parser, instrumentation code is inserted into the source code at predetermined locations (step 108), thereby generating what may be referred to as instrumented source code. In one exemplary embodiment, such instrumentation code comprises counters placed in the front of the expressions, wherein when the compiled expression is executed, the counter will be incremented for each GV mentioned in the expression. Preferably, the instrumentation code is operable in such a manner as to not alter the functionality of the original expression. An exemplary counter instrumented expression can be as follows:
`if(nmpinfo_GV++,nmpinfo > ...),` where the `nmpinfo_GV++` counter is operable to keep track of accesses to the GV

09841931-042501

nmpinfo in the expression. In another exemplary instrumentation code implementation, counters may be inserted to measure function call frequency, rather than measuring GV access frequency. Additional variations of the instrumentation code include, for example, code for tracking the frequency of use of multiple code paths in the source code, for measuring GV accesses from a particular module of the source code, et cetera. If a module-specific instrumentation is used, an exemplary instrumentation counter can be as follows: *if(boot_machdep_nmpinfo_GV++,nmpinfo > ...)*, where the *nmpinfo_GV++* counter is operable with respect to accesses from the module *boot_machdep*.

The instrumented source code generated as the output of the parser operations can be compiled and linked into an executable version of the computer program (step 110). Thereafter, suitable test loads can be run with the executable version of the instrumented source program on an appropriate hardware platform (or, on an architectural simulator if the hardware for running the source program is not available) for exercising the different parts of the source code (step 112).

Referring now to FIG. 2, depicted therein is an exemplary high level functional block diagram of a source code modification system 200 provided in accordance with the teachings of the present invention. A pre-processor block 202 is provided for pre-processing the source code so as to remove any macros, comments, and the like as explained hereinabove. Means such as a parser 204 is operably coupled to the pre-processor block 202 for scanning the source code to recognize select syntax structures therein. The parser structure 204 includes a modification code structure for

instrumenting the source code selectively and automatically. A linker/compiler block 206 is provided as part of the source code modification system 200 for compiling and linking the instrumented source code into an executable version. As
5 alluded to hereinabove, a suitable hardware platform or simulator 208 is provided for running the executable version of the instrumented source code using different test loads.

FIG. 3A depicts an exemplary embodiment of a source code
10 portion 300A prior to instrumentation modification. Two instrumentation points 302A and 302B are illustrated herein. The parser block operable with the source code portion 300A in accordance with the teachings of the present invention includes a code modification portion associated with each of
15 these two instrumentation points. FIG. 3B depicts the exemplary source code portion 300B after instrumentation modification. Two instrumentation code portions 304A and 304B are inserted at instrumentation points 302A and 302B, respectively, based on the code modification portions of the parser employed.

20 It should be appreciated that the source code modification scheme described hereinabove provides an instrumentation software tool that can be customized during the insertion process to include information specific to the point of insertion. In the presently preferred exemplary
25 implementation, the instrumentation tool of the present invention comprises a system for monitoring GV access frequency in the HP-UX Operating System kernel under different test loads. GV counter data for the different test loads can therefore be obtained advantageously for purposes
30 of analyzing the kernel performance and for providing appropriate diagnostic statistics. In an exemplary

embodiment, the GV instrumentation tool optimized for the HP-UX kernel is comprised of the following files: (i) a PERL script *gvcc64* file which changes the build environment by replacing the build environment wrapper for the C compiler; (ii) a PERL script *gvcount* file for automating the procedure by taking the source file list and performing a *clearmake* process; (iii) a C language *gvparse* file optimized for inserting GV counter instrumentation; (iv) a PERL script *gvsort* file for sorting the GV counter data collected by GV name; and (v) a generic C language *parse* file for instrumenting based on specific implementation. Preferably, the *gvsort* file for sorting the GV counter data operates by taking the raw data and producing both module-specific and non-module-specific counts.

FIG. 4 depicts a block diagram of an exemplary multiprocessor (MP) hardware platform 400 operable to run the executable version of an instrumented computer program, e.g., the HP-UX Operating System kernel. Alternatively, the hardware platform 400 is a target processor environment simulated in an architectural simulator wherein the instrumented computer program can be executed in accordance with the teachings of the present invention. Reference numerals 402-1 through 402-N refer to a plurality of processor complexes interconnected together via a high performance, MP-capable bus 404. Each processor complex, e.g., processor complex 402-2, is comprised of a central processing unit (CPU) 406, a cache memory 408, and one or more coprocessors 410. Preferably, the MP system is architected as a tightly coupled SMP system where all processors have uniform access to a main memory 412 and any

input/output (I/O) device 414 in a shared fashion. As an SMP platform, each processor has equal capability to enable any kernel task to execute on any processor in the system. Whereas threads may be scheduled in parallel fashion to run on more than one processor complex, a single kernel controls all hardware and software in an exemplary implementation of the MP system 400, wherein locking and synchronization strategies provide the kernel the means of controlling MP events.

Continuing to refer to FIG. 4, each processor complex is preferably provided with its own data structures, including run queues, counters, time-of-day information, notion of current process(es) and priority. Global data structures available for the entire MP system 400 are protected by means such as semaphores and spinlocks. Furthermore, in other implementations of the MP system, the processors may be arranged as "cells" wherein each cell is comprised of a select number of processors (e.g., 4 processors), interrupts, registers and other resources.

Based upon the foregoing Detailed Description, it should be readily apparent that the present invention provides an efficient scheme for selectively and automatically modifying the source code of a computer program without the shortcomings and drawbacks of the state-of-the-art solutions as set forth in the Background section of the present patent application. The recursive-descent C parser provided in accordance herewith is operable to scan the entire source of the input software and to insert code at all points where instrumentation data needs to be collected. Further, the source code modification scheme allows the source program being instrumented to run at very nearly full speed, thereby

allowing the program to be exercised with heavy workloads and under realistic conditions.

Instrumentation data obtained in the practice of the present invention can be useful in analyzing the performance of any large, complex software program such as an operating system. Further, optimized interfaces can be redesigned at the kernel level of the operating system by collecting appropriate instrumentation data from well defined source code locations.

It is believed that the operation and construction of the present invention will be apparent from the foregoing Detailed Description. While the system and method shown and described have been characterized as being preferred, it should be readily understood that various changes and modifications could be made therein without departing from the scope of the present invention as set forth in the following claims.